An Evolutionary Algorithm for Network Topology Design

Habib Youssef Sadiq M. Sait Salman A. Khan

King Fahd University of Petroleum and Minerals
Department of Computer Engineering
Dhahran-31261, Saudi Arabia
e-mail: {youssef,sadiq,salmana}@ccse.kfupm.edu.sa

Abstract

The topology design of campus networks is a hard constrained combinatorial optimization problem, dictated by physical and technological constraints and must optimize several objectives. Furthermore, due to the nondeterministic nature of network traffic and other design parameters, the objective criteria are imprecise. Fuzzy Logic provides a suitable mathematical framework in such a situation. In this paper, we present an approach based on Simulated Evolution (SE) algorithm for design of campus network topology. Three variations of the algorithm have been presented and compared together. Results show that the third variation, namely, Simulated Evolution with Tabu Search characteristics gives best result.

Keywords: Fuzzy Logic, Campus Networks, Simulated Evolution, Tabu Search, Combinatorial Optimization.

1 Introduction

A typical campus network consists of an interconnected collection of a relatively large number of nodes. The network nodes fall into two general categories: *end-user nodes* which represent network access points consisting of workstations, personal computers, printers, mainframe computers, etc., and the *network active elements* consisting of various devices such as multiplexers, hubs, switches, routers, and gateways. The active elements and links provide the needed physical communication paths between every pair of end-user nodes.

A good network topology is governed by several constraints. Geographical constraints dictate the breakdown of such internetwork into smaller parts or groups of nodes, where each group makes up what is called a LAN. A LAN consists of all the elements that create a networked system up to a router. A campus network is usually made up of a collection of interconnected LANs. Further, the nodes of a LAN may be subdivided into smaller parts, called *LAN segments*, to satisfy other constraints and objectives, for example, minimization of delay, containment of broadcast traffic, and minimization of cabling and equipment cost [1]. The topology design of LAN itself consists of two main issues: *segmentation*, where LAN segments are defined, and *design of actual topology*, which consists of interconnecting the individual segments. Topology design at LAN level usually consists of interconnecting the LAN segments via bridges and layer 2 switches [2].

Following the three-layer hierarchy for strcutred network design [3], the design of such a campus network can be approached in four steps:

1. Assignment of end-user nodes/stations to LAN segments.
2. Assignment of LAN segments to local sites that will make up a single LAN.
3. Design of the internal structure of each local site (i.e., in what topology the LAN segments of a local site are connected). This step serves also to select appropriate switching equipment.
4. Backbone design, where the local sites are connected to the backbone. This step also will dictate the required backbone equipment.

In this work, we have used simulated evolution (SE) algorithm [4, 5] for topology design of structured campus networks based on criteria such as monetary cost, hop count between any source-destination pair, and average network delay per packet. We have confined ourselves to the fourth step, which is the backbone design. For other steps, interested readers can refer to [6]. According to recommended structured cabling standards, the network topology is constrained to be a tree. Hence we target to find a tree topology of desirable quality with respect to the three design objectives.

Since the backbone design problem is a multi-objective combinatorial optimization problem, we resort to fuzzy logic to formulate the various objectives in the form of fuzzy rules that will guide the search toward solutions of desirable quality.

0-7803-7044-9/01/$10.00 ©2001 IEEE

In Section 2, assumptions and notation are given. Section 3 describes computation of objective values and constraints. Section 4 presents the proposed algorithm. Section 5 gives results and discussion. We conclude in Section 6.

2 Assumptions and Notation

- All hosts have either Ethernet (10 or 100 Mbps) or Token Ring (4 or 16 Mbps) interfaces.
- The traffic rates generated among pairs of hosts are assumed known.
- Vertical cabling (interconnection of local sites to backbone switches) is implemented with fiber optic cables.
- Horizontal cabling portion (cabling within the work area/local site) is implemented with Category 5 UTP (or STP for Token-Ring).
- There is a user specified limit on the number of network addresses per subnet.
- Maximum allowed utilization of any link should not exceed a desired threshold (e.g. 60 %).

For the following sections, we shall use the notation given below:

n	number of clusters/local sites.
m	number of LAN segments in a cluster.
T	$n \times n$ local site topology matrix where $t_{ij} = 1$, if local sites i and j are connected and $t_{ij} = 0$ otherwise.
λ_i	traffic on link i.
$\lambda_{max,i}$	capacity of link i.
L	number of links of the proposed topology.
D_{nd}	average delay between any source destination pair.
P_i	maximum number of clusters which can be connected to device i.
γ_{ij}	external traffic between clusters i and j.
γ	overall external traffic.

3 Problem Statement

We seek to find a feasible topology of near optimum *overall cost* with respect to three objectives mentioned earlier. Three important constraints are considered.

1. The first set of constraints is dictated by bandwidth limitation of the links. A good network would be one in which links are "reasonably" utilized, otherwise this would cause delays, congestion, and packet loss. Thus the traffic flow on any link i must never exceed a threshold value:

$$\lambda_i < \lambda_{max,i} \quad i = 1, 2, ..., s \tag{1}$$

where s is the total number of links present in the topology.

2. The second constraint is that the number of clusters attached to a network device i must not be more than the port capacity P_i of that device.

$$\sum_{j=1}^{n} t_{ij} < P_i \quad i = 1, 2, ..., n \qquad \forall i \neq j \tag{2}$$

3. The third set of constraints express the designer's desire to enforce certain hierarchies on the network devices. For example, one might not allow a hub to be the parent of a router or backbone device.

Below, we describe the objective criteria used to measure the goodness of a given topology.

3.0.1 Monetary cost

The cost needs to be minimized; this includes costs of cable and network devices:

$$cost = (l \times c_{cable}) + (c_{nd}) \tag{3}$$

where l represents the total length of cable, c_{cable} represents the cost per unit of the cable used, and c_{nd} represents the combined costs of all the routers, switches, and hubs used.

3.0.2 Average Network Delay

To devise a suitable function for average network delay, we approximate the behavior of a link and network device by an M/M/1 queue and use the formula derived in [2], which suggests that the total average network delay is composed of delays of links and network devices and is given by:

$$D = \frac{1}{\gamma} \sum_{i=1}^{L} \frac{\lambda_i}{\lambda_{max,i} - \lambda_i} + \frac{1}{\gamma} \sum_{i=1}^{d} \sum_{j=1}^{d} \gamma_{ij} B_{ij} \tag{4}$$

where d is the total number of network devices.

3.0.3 Maximum hops

The maximum number of hops between any source-destination pair is also another objective to be optimized. A hop is counted as the packet crosses a network device.

4 Fuzzy SE for Topology Design

4.1 Simulated Evolution

Simulated Evolution (SE) is a general iterative heuristic proposed in [5]. The SE starts with a randomly or constructively generated valid initial solution. A solution is seen as a set of movable elements, each with

an associated goodness measure in the interval [0,1]. The main loop of the algorithm consists of three steps: **evaluation**, **selection** and **allocation**. These steps are carried out repetitively until some stopping condition is satisfied. In the evaluation step, the goodness of each element is estimated. In the selection step, a subset of elements are selected and removed from current solution. The lower the goddness of a particular element, the higher is its selection probability. A bias parameter B is used to compensate for inaccuracies of goodness measure. Finally, the allocation step tries to assign the selected elements to better locations. Other than these three steps, some input parameters for the algorithm are set in an earlier step known as **initialization**.

4.2 Proposed Algorithm

This section describes our proposals of fuzzification of different stages of the SE algorithm. We confine ourselves to tree design because they are minimal and provide unique path between every pair of local sites.

4.3 Initialization

The initial spanning tree topology is generated randomly, while keeping into account the feasibility constraints mentioned earlier.

4.4 Proposed Fuzzy Evaluation Scheme

The **goodness** of each individual is computed as follows. An individual is a **link** which interconnects two local sites. In the *fuzzy evaluation scheme*, monetary cost and optimum depth of a link (with respect to the root) are fuzzy variables. Then the goodness of a link is characterized by:

Rule 1: IF a link is *near optimum cost* AND *near optimum depth* **THEN** it has *high goodness*.

Here, *near optimum cost*, *near optimum depth*, and *high goodness* are linguistic values for the fuzzy variables cost, depth, and goodness. Using and-like compensatory operator [7], Rule 1 translates to the following equation for the fuzzy goodness measure of a link l_i.

$$g_{l_i} = \mu^e(l_i) = \alpha^e \times min(\mu_1^e(l_i), \mu_2^e(l_i)) + (1-\alpha^e) \times \frac{1}{2}\sum_{i=1}^{2} \mu_i^e(l_i) \qquad (5)$$

The superscript e stands for **evaluation**. In Equation 5, $\mu^e(l_i)$ is the fuzzy set of *high goodness links* and α^e is a constant. The $\mu_1^e(l_i)$ and $\mu_2^e(l_i)$ represent memberships in the fuzzy sets *near optimum monetary cost* and *near optimum depth*.

The membership of a link with respect to *near optimum monetary cost* is found using the cost matrix, which gives the costs of each possible link. The cost matrix gives minimum (LCostMin) and maximum (LCostMax) costs among all the link costs. We then find the membership of a link with respect to these bounds. Furthermore, we have normalized the monetary cost with respect to "LCostMax". The required membership function is represented as depicted in Figure 1, where $x - axis$ represents $\frac{LCost}{LCostMax}$, $y - axis$ represents the membership value, $A = \frac{LCostMin}{LCostMax}$, and $B = \frac{LCostMax}{LCostMax} = 1$.

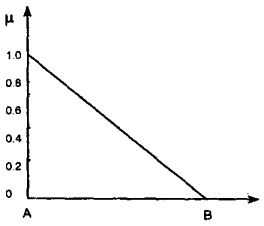

Figure 1: Membership function for the objective to be optimized.

In the same manner, membership of a link with respect to *near optimum depth* is found. The lower limit, "LDepthMin" is taken to be a depth of 1 with respect to the root. The upper bound, "LDepthMax" is taken to be 1.5 times of the maximum depth generated in the initial solution or a maximum of a user specified limit[1], which is taken to be 7, in case if LDepthMax turns out to be more than 7. The membership function with respect to *near optimum depth* is illustrated in Figure 1, where $x - axis$ represents $LDepth$, $y - axis$ represents the membership value, $A = LDepthMin$, and $B = LDepthMax$.

4.5 Selection

Based on the goodness of a link, it is either removed or not. This is done by comparing a random number $RANDOM \in [0,1]$ with $g_{c_i} + B$, where B is the selection bias. Two types of biases, namely, Fixed [4], and Variable [8] are used. The variable bias varies with the average goodness of links in each iteration, but fixed bias is set to a value in the initialization.
the set of links selected for removal.

4.6 Proposed Fuzzy Allocation Scheme

During the **allocation** stage of the algorithm, the selected links are removed from the topology one at a

[1]This user specified limit may be a design constraint, e.g., if each hop represents a router that uses Routing Information Protocol (RIP) then a reasonable limit would be 7, i.e., a branch of the tree should not have more than 7 routers.

time. For each removed link, new links are tried in such a way that they result in overall better solution. In the *fuzzy allocation scheme*, the three criteria to be optimized are combined using fuzzy logic to characterize a good topology. In fuzzy logic, this can easily be stated by the following fuzzy rule:

Rule 2: **IF** a solution X has *low monetary cost* AND *low average network delay* AND *low maximum number* of hops between any source-destination pair **THEN** it is a *good topology.*

The words "low monetary cost", "low average network delay", "low maximum number of hops", and "good topology" are linguistic values, each defining a fuzzy subset of solutions. Each fuzzy subset is defined by a membership function μ which returns a value in the interval [0,1] describing the degree of satisfaction with the particular objective criterion. Using the and-like operator [7], the above fuzzy rule reduces to the following equation.

$$\mu^a(x) = \beta^a \times min(\mu_1^a(x), \mu_2^a(x), \mu_3^a(x)) + (1-\beta^a) \times \frac{1}{3}\sum_{i=1}^{3} \mu_i^a(x) \qquad (6)$$

where $\mu^a(x)$ is the membership value for solution x in the fuzzy set *good topology* and β^a is a constant in the range [0,1]. The superscript a stands for allocation. Here, μ_i^a for $i = \{1,2,3\}$ represents the membership values of solution x in the fuzzy sets *low monetary cost, low average network delay,* and *low maximum number of hops between any source-destination pair* respectively. The solution which results in the maximum value for Equation 6 is reported as the best solution found by the SE algorithm.
Below we will see how to get the membership functions for the three criteria we have mentioned above.

4.6.1 Membership for Monetary Cost

First, we determine two extreme values for monetary cost, i.e., the minimum and maximum values. The minimum value, "TCostMin", is found by using the Esau-Williams algorithm [9], with all the constraints completely relaxed. The maximum value of monetary cost,"TCostMax", is taken to be the monetary cost generated in the initial solution. The monetary cost is normalized with respect to "TCostMax". The corresponding membership function is shown in Figure 1, where $x - axis$ represents $\frac{TCost}{TCostMax}$, $y - axis$ represents the membership value, $A = \frac{TCostMin}{TCostMax}$, and $B = \frac{TCostMax}{TCostMax} = 1$.

4.6.2 Membership For Network Delay

We determine two extreme values for average network delay. The minimum value, "TDelayMin", is found by connecting all the nodes to the root directly, ignoring all the constraints and then calculating the average network delay using Equation 4. The maximum value of average delay,"TDelayMax", is taken to be the average delay generated in the initial solution. The average delay is normalized with respect to "TDelayMax". The membership function is shown in Figure 1, where $x - axis$ represents $\frac{TDelay}{TDelayMax}$, $y - axis$ represents the membership value, $A = \frac{TDelayMin}{TDelayMax}$, and $B = \frac{TDelayMax}{TDelayMax} = 1$.

4.6.3 Membership For Maximum Hops

Again, two extreme values are determined. The minimum value, "THopsMin", is taken to be 1 hop, which will be the minimum possible in any tree. The maximum value,"THopsMax", is taken to be the maximum number of hops between any source-destination pair generated in the initial solution. The membership function is shown in Figure 1, where $x - axis$ represents $THops$, $y-axis$ represents the membership value, $A = THopsMin$, and $B = THopsMax$.
In the proposed allocation scheme, all the selected links are removed one at a time and trial links are placed for each removed link. We start with the head-of-line link, i.e. the link with the worst goodness. We remove this link from the topology. This divides the topology into two disjoint topologies.
Now the placing of trial links begins. At most ten trial moves (i.e., trial links) are evaluated for each removed link. However, some moves may be invalid. Therefore, we search for only four "valid" moves. Whenever we find four valid moves, we stop, otherwise we continue until a total of ten moves are evaluated (whether valid or invalid). The removal of a link involves two nodes P and Q, of which node P belongs to the subtree which contains the root node and node Q belongs to the other subtree. For the ten moves we make, five of them are greedy and five are random. For the greedy moves, we start with node Q and five *nearest* nodes in the other subtree are tried. For the random moves, we select any two nodes in the two subtrees and connect them.
It may so happen that all the ten moves are invalid, in which case the original link is placed back in its position. The valid moves are evaluated based on Equation 6 and the best move among the ten moves is made permanent. This procedure is repeated for all the links that are present in the set of selected links. In the allocation phase, we have used tabu search characteristics. As mentioned above, in the allocation phase

Table 1: Characteristics of test cases used in our experiments. LCostMin, LCostMax, and TCostMin are in US$. TDelayMin is in milliseconds. Traffic is in Mbps.

Name	# of Local Sites	LCostMin	LCostMax	TCostMin	TDelayMin	Traffic
n15	15	1100	9400	325400	2.14296	24.63
n25	25	530	8655	469790	2.15059	74.12
n33	33	600	10925	624180	2.15444	117.81
n40	40	600	11560	754445	2.08757	144.76
n50	50	600	13840	928105	2.08965	164.12

certain number of moves are made for each link in the selection set and the best move is accepted, making the move (i.e., link) permanent. This newly accepted link is also saved in the *tabu list*. Thus our *attribute* is the link itself. The *aspiration criterion* adopted is that if the link that had been made tabu produces a higher membership value than the current one in the membership function "good topology", then we will override the tabu status of the link and make it permanent. This strategy prevents the selection and allocation of a tree from repetitvely removing the same link and replacing it with a link of equal or worse goodness. For details of tabu search, refer to [4].

4.7 Stopping Criterion

Fixed number of iterations is used as stopping criterion. Experimentation with different values of iterations showed that SE algorithm converges within 4000 iterations for all test cases.

5 Results and Discussion

The proposed SE algorithms have been tested on several randomly generated networks. For each test case, the traffic generated by each local site was collected from real sites. Other characteristics, such as the number of ports on a network device, its type, etc. were assumed. However, the costs of network devices and links were collected from vendors. The characteristics of test cases are listed in Table 1. The smallest test network has 15 local sites and the largest has 50 local sites.

Table 2 shows the best solutions generated by best fixed bias SE_FF and SE_VB, while Table 3 compares them. It is clear from these tables that, in general, SE_VB produces comparable results with SE_FF as far as "monetary cost" objective is concerned. For "average network delay" and "maximum hops" objectives, a general trend is that SE_FF performs better than SE_VB. As far as execution time is concerned, SE_VB has lower execution time than best fixed bias SE_FF for smaller cases (such as **n15**, **n25**, and **n33**), while for bigger cases (**n40** and **n50**), SE_VB has higher execution time than SE_FF. However, if we consider the time spent in trial runs of SE_FF algorithm to find the best fixed bias, then SE_VB can be considered better than the fixed bias SE_FF. There were at least 3 trial runs with different bias values to identify the best value for each test case for SE_FF. For SE_VB, there is no need to run several trials. Figures 2(a), (b), and (c) show the progression of the two algorithms with respect to the three optimization objectives.

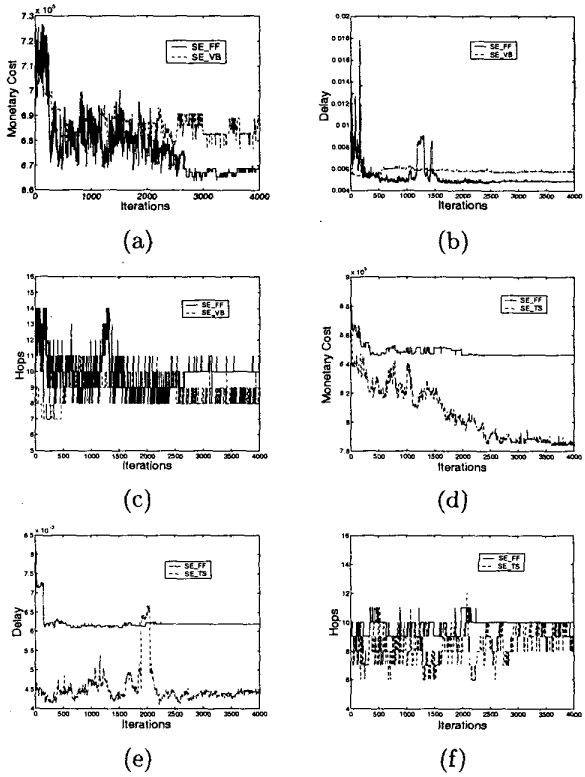

Figure 2: Progression of objective values with iterations for SE_FF and SE_VB for (a) monetary cost (b) delay (c) hops. (d), (e), (f) compare the same objectives respectively for SE_FF and SE_TS.

Table 2 also shows the results of SE_TS. Table 3 shows the percentage improvement achieved by best tabu list size SE_TS when compared to SE_FF. From these tables, it is seen that SE_TS performs better than SE_FF for monetary cost objective. For all test cases, a gain is achieved by SE_TS. Similarly, for average network

Table 2: Comparison of SE_FF, SE_VB, and SE_TS. B = best bias, C = Cost in US $, D = Delay in msec/packet, H = hops, T = execution time (minutes), TL= Tabu list size.

Case	SE_FF					SE_VB				SE_TS				
	B	C	D	H	T	C	D	H	T	TL	C	D	H	T
n15	0.2	314400	3.282	5	4	305500	4.135	7	1	2	297100	2.78	4	2.25
n25	0.2	509050	4.26	7	5	512415	4.37	7	4.4	5	483210	3.537	6	4
n33	0.0	687760	4.729	8	40	702815	5.319	7	17	6	682465	4.19	6	8
n40	0.3	866900	4.126	8	12	800580	6.637	10	42	7	783970	4.441	9	26
n50	0.3	1061900	5.32	9	8	1042080	8.236	10	62	7	983020	5.245	11	65

Table 3: Percentage improvement achieved by SE_VB compared to SE_FF and SE_TS compared to SE_FF. C = Cost in US $, D = Delay in ms/packet, H = hops.

Case	SE_VB vs SE_FF			SE_TS vs SE_FF		
	C	D	H	C	D	H
n15	2.83	-25.99	-40	5.5	15.29	20
n25	-0.657	-2.51	0	5.07	16.97	14.29
n33	-2.19	-12.48	12.5	0.755	11.4	25
n40	7.65	-60.85	25	9.57	-7.63	-12.5
n50	1.87	-54.8	-11.1	7.43	1.41	-22.2

delay metric, SE_TS achieves gain in all cases. For maximum number of hops metric, a gain is achieved for all the cases except *n50*. However, the loss in maximum hops for *n50* is compensated by the improvement in the monetary cost and delay metrics. As far as the execution time is concerned, it is also comparable. Figures 2(d),(e), and (f) show the progression of SE_FF and SE_TS for the three optimization objectives. The reason SE_TS has better performance than SE_FF is the following. In SE_FF, since the search space for valid solutions is limited, it happens that after some iterations, same moves are repeated and the algorithm keeps searching in the same search space most of the time, while in SE_TS, more search space is covered because previous moves remain tabu for some time, causing the algorithm to diversify the search into another subarea.

6 Conclusion

In this paper we have presented three variations of fuzzy SE for backbone topology design of campus networks. The proposed SE algorithms were always able to find feasible topologies with desirable qualities. Comparisons among the variations showed that the solution subspace investigated by SE_TS is of superior quality than that of the other two variations. Further, as time elapsed, all variants progressively evolved toward better solutions, a desirable characteristic of evolutionary heuristics.

Acknowledgments

Authors acknowledge King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia, for all support.

References

[1] Habib Youssef, Sadiq M. Sait, and Osama A. Issa. Computer-Aided Design of Structured Backbones. In *15th National Computer Conference and Exhibition*, pages 595–607, October 1997.

[2] R. Elbaum and M. Sidi. Topological Design of Local-Area Networks Using Genetic Algorithm. *IEEE/ACM Transactions on Networking*, pages 766–778, October 1996.

[3] Habib Youssef, Sadiq M. Sait, and Salman A. Khan. A Fuzzy Simulated Evolution Algorithm For Topology Design of Campus Networks. In *IEEE Congress on Evolutionary Computation*, 2000.

[4] Sadiq M. Sait and Habib Youssef. *Iterative Computer Algorithms and their Application to Engineering.* IEEE Computer Science Press, Dec. 1999.

[5] Ralph Michael Kling. *Optimization by Simulated Evolution and its Application to cell placement.* Ph.D. Thesis, University of Illinois, Urbana, 1990.

[6] Salman A. Khan. *Topology Design of Enterprise Networks.* MS Thesis. King Fahd University of Petroleum and Minerals, 1999.

[7] Ronald Y. Yager. On Ordered Weighted Averaging Aggregation Operators in Multicriteria Decisionmaking. *IEEE Transactions on Systems, Man, and Cybernatics*, 18(1):183–190, Jan 1988.

[8] Habib Youssef, Sadiq M. Sait, and Ali S. Hussain. Adaptive Bias Simulated Evolution Algorithm For Placement. In *ISCAS 2001*, pages 355–358, 2001.

[9] L. R. Esau and K. C Williams. On teleprocessing system design. A method for approximating the optimal network. *IBM System Journal*, 5:142–147, 1966.